mathematics

MDPI

Article

Real-Time Estimation of R_0 for COVID-19 Spread

Theodore E. Simos [1,2,3,*], Charalampos Tsitouras [4], Vladislav N. Kovalnogov [1], Ruslan V. Fedorov [1] and Dmitry A. Generalov [1]

[1] Laboratory of Applied Mathematics for Solving Interdisciplinary Problems of Energy Production, Ulyanovsk State Technical University, Severny Venetz St. 32, 432027 Ulyanovsk, Russia; kvn@ulstu.ru (V.N.K.); r.fedorov@ulstu.ru (R.V.F.); dmgeneralov@mail.ru (D.A.G.)
[2] Scientific and Educational Center "Digital Industry", South Ural State University, 76, Lenin Av., 454080 Chelyabinsk, Russia
[3] Section of Mathematics, Department of Civil Engineering, Democritus University of Thrace, 67100 Xanthi, Greece
[4] General Department, Euripus Campus, National and Kapodistrian University of Athens, 34400 Athens, Greece; tsitourasc@uoa.gr
* Correspondence: simost@susu.ru

Abstract: We propose a real-time approximation of R_0 in an SIR-type model that applies to the COVID-19 epidemic outbreak. A very useful direct formula expressing R_0 is found. Then, various type of models are considered, namely, finite differences, cubic splines, Piecewise Cubic Hermite interpolation and linear least squares approximation. Preserving the monotonicity of the formula under consideration proves to be of crucial importance. This latter property is preferred over accuracy, since it maintains positive R_0. Only the Linear Least Squares technique guarantees this, and is finally proposed here. Tests on real COVID-19 data confirm the usefulness of our approach.

Keywords: SIR epidemic model; COVID-19; ordinary differential equations; splines; least squares

MSC: 92C60; 65L12; 65F20

Citation: Simos, T.E.; Tsitouras, C.; Kovalnogov, V.N.; Fedorov, R.V.; Generalov, D.A. Real Time Estimation of R_0 for COVID-19 Spread. *Mathematics* **2021**, *9*, 664. https://doi.org/10.3390/math9060664

Academic Editor: Alexander Zeifman

Received: 15 February 2021
Accepted: 18 March 2021
Published: 20 March 2021

1. Introduction

The SIR model describes the dynamics of an epidemic. Kermack and McKendrick introduced this in their pioneering work [1]. In this model, we consider a homogeneous population. Thus, no age-structures or group behaviours are taken into account. The population is divided into three homogeneous sections: susceptible people S, infectious people I and recovered people R. Since S, I and R change in time t, we represent these numbers as functions $S(t)$, $I(t)$ and $R(t)$. Births and deaths are ignored, considering a constant population size N within time t i.e., $S(t) + R(t) + I(t) = N$.

We proceed setting $i(t) = I(t)/N$, $s(t) = S(t)/N$, $r(t) = R(t)/$, i.e., the corresponding rates.

The following system of ordinary differential equations describes the SIR model [2–4]

$$\begin{aligned} \frac{ds(t)}{dt} &= -\beta(t)i(t)s(t), \\ \frac{di(t)}{dt} &= \beta(t)i(t)s(t) - \gamma(t)i(t), \\ \frac{dr(t)}{dt} &= \gamma \cdot i(t). \end{aligned} \tag{1}$$

The function $\beta(t)$ is the transmission with respect to time, while $\gamma(t)$ is the rate of recoveries. Ideally, $\beta(t)$ and $\gamma(t)$ are constants. Notice that

$$\frac{ds(t)}{dt} + \frac{di(t)}{dt} + \frac{dr(t)}{dt} = 0,$$

and consequently

$$s(t) + i(t) + r(t) = 1 = s(0) + i(0) + r(0).$$

A very important quantity is the reproduction number, given as

$$R_0(t) = \frac{\beta(t)}{\gamma(t)}.$$

Ordinary Differential Equations (1) can be solved numerically using Runge–Kutta pairs [5–7]. One of the main difficulties arising is the estimation of parameters $\beta(t)$, $\gamma(t)$ and, consequently, of $R_0(t)$.

There is an ongoing interest in SIR-type models. Bertrand and Pirch [8] presented a least-squares finite element method for the SEIQRD model. Cherniha and Davydovych [9] studied a nonlinear model based on logistic equations, while Keller et al. [10] simulated the spread of an infectious disease across a heterogeneous and continuous landscape. E. Kuhl [11] focused on modeling various stages of COVID-19 outbreak and Viguerie et al. [12] simulated COVID-19 via an SEIRD model.

From (1) we deduce

$$\frac{ds}{dr} = -\frac{\beta(t)}{\gamma(t)} s,$$

and, consequently, we arrive at

$$R_0 = -\frac{1}{s}\frac{ds}{dr}. \tag{2}$$

Thus, our concern here is actually a reliable estimation of $\frac{ds}{dr}$.

2. Interpolating the Past

We proceed to set t_n as the current time, while $s_n = s(t_n)$ and $r_n = r(t_n)$ are the current observed numbers of susceptible and recovered. Consequently, t_{n-1} corresponds to the previous time step, usually the previous day, i.e., $t_{n-1} = t_n - 1$ and all t are considered integers. Past values of s_{n-1}, r_{n-1}, s_{n-2}, $r_{n-2}, \cdots$ and population N are usually known and we seek an approximation of the $R_0(t)$ trough $\frac{ds}{dr}$. Through the estimation of this derivative at the right (current) point, $\frac{ds_n}{dr_n}$. In the following, it is convenient to consider s as a function with respect to the increasing variable r. We have various choices.

2.1. Backward Finite Differences

The celebrated Taylor series [13] states that

$$s(r+h) = s(r) + hs'(r) + \frac{1}{2!}h^2 s''(r) + \frac{1}{3!}h^3 s'''(r) + \frac{1}{4!}h^4 s^{(4)}(r) + \cdots, \tag{3}$$

We are given various values of s at distinct points, $r_0 < r_1 < \cdots < r_n$ and name them for simplicity $s_0 = s(r_0)$, $s_1 = s(r_1), \cdots$, etc. Backward finite differences answer the question of approximating the derivatives of s at r_n. Thus, we may derive

$$\frac{ds_n}{dr_n} = s'_n \approx \frac{s_n - s_{n-1}}{r_n - r_{n-1}} \tag{4}$$

which is easily verified from (3) using its implicit Euler variant

$$s_n = s(r+h) \approx s(r) + hs'(r+h) = s_{n-1} + hs'_{n-1},$$

with $h = r_n - r_{n-1}$. Approximation (4) is said to be the first order of accuracy, since no higher orders of h are involved in using (3), e.g., h^2, $h^3, \cdots$ etc. After substituting $-\frac{1}{s}$ with the mean value $-\frac{2}{s_{n-1}+s_n}$, we obtain the formula

$$R_0 \approx -\frac{2}{s_{n-1}+s_n} \cdot \frac{s_n - s_{n-1}}{r_n - r_{n-1}}. \tag{5}$$

This latter estimation (5) is of limited value, since it is based on data from two days and fluctuates intensively. Thus, we may proceed to higher-order differences. A second-order backward finite difference approximation of $\frac{ds_n}{dr_n}$ produces the formula

$$R_0 \approx -\frac{3}{s_{n-2}+s_{n-1}+s_n} \cdot \begin{pmatrix} \frac{(r_n - r_{n-1})}{(r_{n-2}-r_{n-1})(r_{n-2}-r_n)} \cdot s_{n-2} \\ + \frac{(r_{n-2}-r_n)}{(r_{n-2}-r_{n-1})(r_{n-1}-r_n)} \cdot s_{n-1} \\ - \frac{(r_{n-2}+r_{n-1}-2r_n)}{(r_{n-2}-r_n)(r_{n-1}-r_n)} \cdot s_n \end{pmatrix}. \tag{6}$$

As we worked in (5), we substituted $-\frac{1}{s}$ with the mean value of s_{n-2}, s_{n-1} and s_n. We observe that s is clearly a decreasing variable. Then, $\frac{ds_n}{dr_n} \leq 0$ and $R_0 \geq 0$ must hold. In (5), this property is preserved. However, this is not true for (6). Let us check this using a small example. Take the following artificial data

$$\begin{array}{lll} r_{n-2} = 0.001, & r_{n-1} = 0.002, & r_n = 0.003, \\ s_{n-2} = 0.996, & s_{n-1} = 0.9915 & \text{and } s_n = 0.991 \end{array} \tag{7}$$

to verify that, according to (6), we get $R_0 \approx -1.5108 < 0$!

This happens because second- and higher-order backward finite differences do not preserve monotonicity [14]. Dealing with higher-order methods is,therefore, of no meaning.

Notice that using (5), we have $R_0 \approx 0.505$ for the last two days (i.e., based on the r-interval $[0.002, 0.003]$). However, the previous day's estimation was $R_0 \approx 4.539$. This oscillation is also unacceptable.

2.2. Cubic Splines

Cubic splines [15] are a very interesting tool that can be used in various fields [16,17]. The interval $[r_0, r_n]$ is divided to n subintervals $[r_0, r_1]$, $[r_1, r_2], \cdots, [r_{n-1}, r_n]$. Using cubic splines, we can obtain n polynomials of third degree, with each one active in every subinterval. Concentrating again on the interval $[r_{n-2}, r_n]$, we may obtain the following formula after using the software found in [18]

$$R_0 \approx -\frac{3}{s_{n-2}+s_{n-1}+s_n} \cdot \begin{pmatrix} \frac{(r_n - r_{n-1})}{2(r_{n-1}-r_{n-2})(r_n-r_{n-2})} \cdot s_{n-2} \\ + \frac{(2r_{n-2}-r_{n-1}-r_n)}{2(r_{n-2}-r_{n-1})(r_{n-1}-r_n)} \cdot s_{n-1} \\ + \frac{(2r_{n-2}+r_{n-1}-3r_n)}{2(r_{n-2}-r_n)(r_n-r_{n-1})} \cdot s_n \end{pmatrix}.$$

The data (7) produce $R_0 \approx -0.5036 < 0$, which is unacceptable. For this dataset, we can obtain two polynomials of the third degree. In the interval $[0.002, 0.003]$, the polynomial approximation of s is

$$s(r) \approx -10^6(r - 0.002)^3 + 3000(r - 0.002)^2 - 2.5(r - 0.002) + 0.9915,$$

with the positive derivative $s'(0.003) \approx 0.5 > 0$ at the rightmost point. Through this counterexample, we deduce that cubic splines do not preserve monotonicity either.

2.3. Piecewise Cubic Hermite Interpolant

An alternative to cubic splines is Piecewise Cubic Hermite interpolation (PCHIP). In cubic splines, the coincidence of higher derivatives at the nodes is demanded. In PCHIP,

this is abandoned in order to achieve monotone cubic polynomials for monotone data. The issue is rather complicated to explain here. More details can be found in [19] or in the MATLAB function `pchip`.

For the data (7), we can obtain the following polynomial active in the interval $r \in [0.002, 0.003]$

$$s(r) \approx 10^5(r-0.002)^3 + 300(r-0.002)^2 - 0.9(r-0.002) + 0.9915, \tag{8}$$

with derivative $s'(0.003) = 0$ at the rightmost point. Monotonicity is marginally preserved, but $R_0 = 0$ in this case. Even if we add more points from the past, we will not change the situation. The method tries always to produce a monotone polynomial of the third degree. Thus, it will make very small changes to (8), regardless of how many points we add from the past.

All of the above three type of approximation (2.1–2.3) may attain higher algebraic orders, i.e., second-order or higher. This means that we may obtain better accuracy in the approximation of $\frac{ds}{dr}$. However, these actually fail, since they do not preserve monotonicity, which is by no means a property of the function $s(r)$ and is also present in the corresponding data. Failure to preserve monotonicity is catastrophic. We may sometimes experience $R_0 < 0$ then.

2.4. Linear Least Squares

Finally, we propose estimation $\frac{ds_n}{dr_n}$ by the slope of linear least squares approximation of the data in $[r_0, r_n]$. Then, we get

$$R_0 \approx -\frac{n+1}{\sum_{j=0}^{n} s_j} \cdot \frac{(n+1)\sum_{j=0}^{n} r_j s_j - (\sum_{j=0}^{n} r_j)(\sum_{j=0}^{n} s_j)}{(n+1)\sum_{j=0}^{n} r_j^2 - (\sum_{j=0}^{n} r_j)^2}. \tag{9}$$

The denominator of (9) is always positive, since r is ascending.

For the special case of using data in the interval $[r_{n-2}, r_n]$, we have

$$R_0 \approx -\frac{3}{s_{n-2}+s_{n-1}+s_n} \cdot \left\{ \frac{\begin{pmatrix} 3(r_{n-2}s_{n-2} + r_{n-1}s_{n-1} + r_n s_n) \\ -(r_{n-2}+r_{n-1}+r_n)(s_{n-2}+s_{n-1}+s_n) \end{pmatrix}}{3(r_{n-2}^2 + r_{n-1}^2 + r_n^2) - (r_{n-2}+r_{n-1}+r_n)^2} \right\}. \tag{10}$$

After using (10) with data (7), we arrive at an acceptable $R_0 \approx 2.523$.

It seems that (9) furnishes a balanced result in view of the corresponding results found by (5). The reason for the final choice of this method is the preservation of monotonicity, which helps to avoid unpleasant outcomes such as $\frac{ds}{dr} > 0$. This is true, since the slope of a least squares line applied on decaying data is obligatorily negative. The latter does not always happen in cases where, e.g., a parabola passes through these points.

The question that raises now is the size of the data used. i.e., $n =$?. A strategy to address this is begins with $n = 2$ and estimates R_0 with a value named R_0^2. Continue with $n = 3, 4, \cdots$ and estimates R_0^3, $R_0^4, \cdots$, respectively. We stop the iteration whenever two consecutive estimations of R_0 differ less than $\frac{1}{100}$. That is, whenever

$$|R_0^k - R_0^{k-1}| < 0.01,$$

we accept R_0^k as the value on demand. In case this does not happen, we accept R_0^{21} as the final approximation of R_0. We implement this procedure as a MATLAB [20] program in the Appendix A.

3. Preliminary Tests

We choose the static case $\beta(t) = \beta = 0.1$ and $\gamma(t) = \gamma = 0.05$ with initial conditions $s(0) = s(t_0) = s_0 = 0.9999$, $i_0 = 0.0001$ and $r_0 = 0$. Here, $R_0 = 2$. Then, we obtain the

values of s, i and r for $t_1, t_2, t_3, t_4, \cdots, t_{100}$ by the following lines in the command window of MATLAB.

```
>> beta=0.1;gamma=0.05;
>> fcn=@(t,x)[-beta*x(1)*x(2);
beta*x(1)*x(2)-gamma*x(2);
gamma*x(2)];
>> [tout,xout]=ode45(fcn,(0:1:100)',[0.9999 0.0001 0]');
```

Now, we may approximate the actual value of R_0 for $t_{20} = 20, t_{21} = 21, \cdots, t_{100} = 100$. Thus, we type

```
>> ro=zeros(101,1);
>> j1=21:101,ro(j1)=r00(xout(j1-20:j1,3),xout(j1-20:j1,1));end;
>> max(abs(ro(21:101)-2))
ans =
3.2964e-005
```

i.e., we obtain almost five digits of accuracy. This result was obtained by using the data from only tree consecutive data pairs (i.e., by using only (10)). We observed this behavior since the parameters are constant. The same result is also attained for other selections of β and γ.

The method also applies in case of varying parameters. In the next test, we choose constant $\gamma = 0.1$ and $\beta = 0.1 + \frac{t}{1000}$. Then, we type in MATLAB

```
>> fcn=@(t,x) [-(0.1+t/1000)*x(1)*x(2);
(0.1+t/1000)*x(1)*x(2)-.1*x(2);
0.1*x(2)];
>> [tout,xout]=ode45(fcn,(0:1:100)',[0.999 0.001 0]');
```

In this paradigm, we have $R_0 = 1 + \frac{t}{100}$, and the performance of the method is checked by the following

```
>> ro=zeros(101,1);
>> for j1=21:101,ro(j1)=r00(xout(j1-20:j1,3),xout(j1-20:j1,1));end;
>> max(abs(ro(21:101)-1-(20:100)'./100))
ans =
0.0149
```

i.e., the error is in the second decimal point. This is a rather good approximation of a R_0, which takes values in the interval $[1, 2]$.

A very interesting issue is that using mean value of data s in (2) calibrates the result to the correct value of R_0. The percentage of susceptible s varies slowly. Using the mean value instead of s_n causes only a small difference, which delivers 2–3 more digits of accuracy. Additionally, we mention that this new approach is much easier to compute than our previous method [21]. It also seems to achieve better accuracy.

4. Tests on Real COVID-19 Data

We will test the new approach using real COVID-19 data. The time-series data of confirmed, recovered, and death cases for various countries were retrieved from [22]. The data are presented in the format shown in Table 1.

The two rightmost columns sum to form vector r, after dividing this sum by the country's population. The vector s is formed by the division of confirmed cases by population. The population of each country was retrieved from Wikipedia [23]. The data in [22] are not always reliable. Data are missing or reported inaccurately for some countries. The method presented here does not apply properly at the outbreak of a disease, when values r are rather small. Then, we may apply some scale, e.g., use vectors S and R. The values in r

have to vary somehow in order to obtain reliable results. Thus, no trustworthy results can be derived from the following Table 2.

Table 1. Data as presented in [22].

Date	Country	Confirmed	Recovered	Deaths
...				
6 February 2021	Russia	3,907,653	3,398,545	75,010
7 February 2021	Russia	3,923,461	3,418,329	75,430
8 February 2021	Russia	3,939,162	3,434,163	75,828
9 February 2021	Russia	3,953,970	3,455,582	76,347
10 February 2021	Russia	3,968,228	3,477,760	76,873
11 February 2021	Russia	3,983,031	3,499,230	77,415
12 February 2021	Russia	3,997,898	3,519,689	77,911
...				

Table 2. Data for Russia at the beginning of COVID-19 outbreak, as presented in [22].

Date	Country	Confirmed	Recovered	Deaths
...				
1 March 2020	Russia	2	2	0
2 March 2020	Russia	3	2	0
3 March 2020	Russia	3	2	0
4 March 2020	Russia	3	2	0
5 March 2020	Russia	4	2	0
6 March 2020	Russia	13	2	0
7 March 2020	Russia	13	2	0
...				

We do not believe that any serious method can extract something reliable from the above data. By this, we mean that, at the beginning of an outbreak, we may not obtain an explicit picture of the situation from a sole country's data. Only a global view may raise some interest in these numbers. Thus, the method at hand may apply after the initial development of the pandemic.

We present the results for Russia in Figure 1, for Germany in Figure 2, and for Italy in Figure 3.

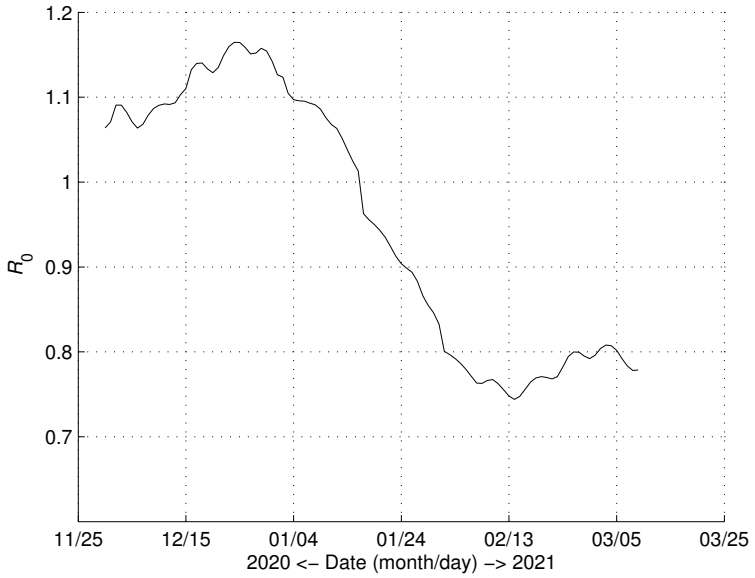

Figure 1. R_0 for Russia from 30 November 2020 to 9 March 2021.

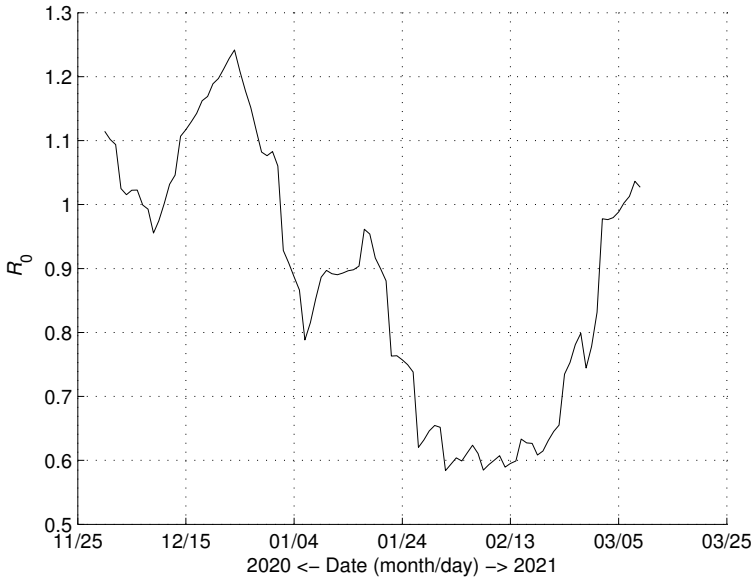

Figure 2. R_0 for Germany from 30 November 2020 to 9 March 2021.

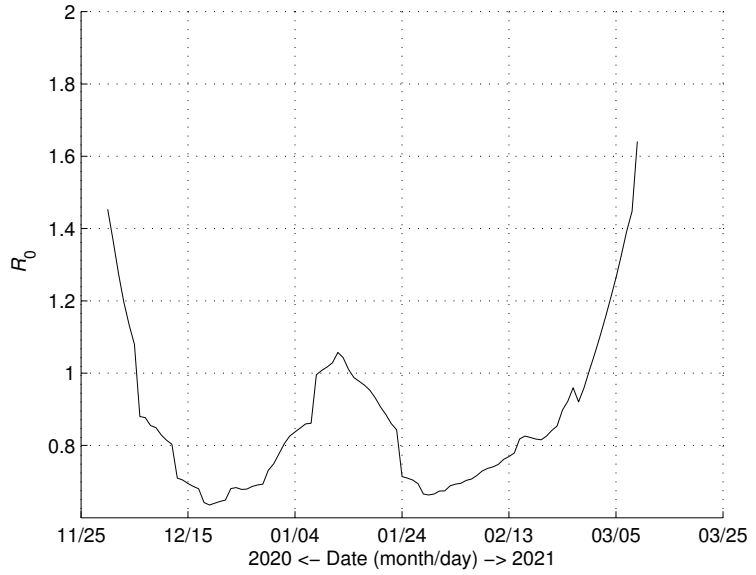

Figure 3. R_0 for Italy from 30 November 2020 to 9 March 2021.

We produced these results using at least 14 of the latest data in order to obtain smoother curves.

We observe that R_0 decays for Russia and, after mid-January 2021, stays below 1. The corresponding parameter for Germany stayed below 1 for 2021, but it seems that this was not true from the beginning of March. Finally, R_0 stayed below 1 for Italy until the end of February, and climbed above it in March.

The data for the countries mentioned above seem to be reliable. However, there are cases with misreported data. In France, after a year, only about a quarter of million were reported as recovered, while there are 4 million infected! The same problem is apparent in the data for the UK and other countries.

Thus, the new method cannot apply to corrupted data, as expected for any other method. Least squares may circumvent an outlier or some misreported data, but consistently faulty data are untreatable.

5. Conclusions

A new formula for directly estimating R_0 present in the SIR epidemic model is derived here. Using only the percentage values of susceptible s, and recovered r at consecutive days, we form a linear least squares approximation of the derivative $\frac{ds}{dr} \leq 0$. This approximation is non-positive (i.e., preserves monotonicity). Then, the new formula stays close enough to R_0. For use with real COVID-19 data, we implemented an iterative technique that promises convergence to the actual value of R_0. Similar research is planned in the future for other models, such as SIS, SIRD, and SEIR.

Author Contributions: All authors (T.E.S., C.T., V.N.K., R.V.F. and D.A.G.) contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: This research received external funding: mega grant of the Government of the Russian Federation, project No. 2020-220-08-6251.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data were retrieved from [22].

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

The MATLAB [20] function r00 that implements the proposed iterative scheme of successive linear least squares.

```
%------------------------------------------------------------------------
function roo=r00(r,s);
% give r, s vectors with 21 entries
j1=1;roold=1000;r=r(:);s=s(:);
roo=r0(r(end-j1:end),s(end-j1:end));
while abs(roo-roold)>0.01 && j1<20,
roold=roo;
j1=j1+1;
roo=r0(r(end-j1:end),s(end-j1:end));
end;
return;

function ro=r0(r,s);
% least squares estimation of r0 through formula
%       1 ds
%  r0=- -*--
%       s dr
%
if (length(r)~=length(s)) | (length(r)<2),
then disp('vectors of improper length');return;
end;

dsdr=(length(r)*sum(r.*s)-sum(r)*sum(s))/(length(r)*sum(r.^2)-(sum(r))^2);
ro=-1/mean(s(1:end))*dsdr;
return;
%------------------------------------------------------------------------
```

References

1. Kermack, W.O.; McKendrick, A.G. Contributions to the mathematical theory of epidemics—I. *Proc. R. Soc. A* **1927**, *115*, 700–721.
2. Beckett, S.J.; Dominguez-Mirazo, M.; Lee, S.; Andris, C.; Weitz, J.S. Spread of COVID-19 through Georgia, USA. Near-term projections and impacts of social distancing via a metapopulation model. *MedRxiv* **2020**. [CrossRef]
3. Korsbo, N.; Jönsson, H. It's about time: Analysing simplifying assumptions for modelling multi-step pathways in systems biology. *PLoS Comput. Biol.* **2020**, *16*, e1007982. [CrossRef] [PubMed]
4. Weitz, J.S.; Beckett, S.J.; Coenen, A.R.; Demory, D.; Dominguez-Mirazo, M.; Dushoff, J.; Leung, C.-Y.; Li, G.; Măgălie, A.; Park, S.W.; et al. Modeling shield immunity to reduce COVID-19 epidemic spread. *Nat. Med.* **2020**, *26*, 849–854. [CrossRef] [PubMed]
5. Tsitouras, C. Optimized explicit Runge-Kutta pair of orders 9(8). *Appl. Numer. Math.* **2001**, *38*, 123–134. [CrossRef]
6. Tsitouras, C. Runge–Kutta pairs of order 5(4) satisfying only the first column simplifying assumption. *Comput. Maths Appl.* **2011**, *62*, 770–775. [CrossRef]
7. Tsitouras, C.; Papakostas, S.N. Cheap Error Estimation for Runge-Kutta pairs. *SIAM J. Sci. Comput.* **1999**, *20*, 2067–2088. [CrossRef]
8. Bertrand, F.; Pirch, E. Least-Squares Finite Element Method for a Meso-Scale Model of the Spread of COVID-19. *Computation* **2021**, *9*, 18. [CrossRef]
9. Cherniha, R.; Davydovych, V. A Mathematical Model for the COVID-19 Outbreak and Its Applications. *Symmetry* **2020**, *12*, 990. [CrossRef]
10. Keller, J.P.; Gerardo-Giorda, L.; Venezianic, A. Numerical simulation of a susceptible–exposed–infectious space-continuous model for the spread of rabies in raccoons across a realistic landscape. *J. Biol. Dyn.* **2013**, *7*, 31–46. [CrossRef] [PubMed]
11. Kuhl, E. Data-driven modeling of COVID-19—Lessons learned. *Extrem. Mech. Lett.* **2020**, *40*, 100921. [CrossRef] [PubMed]
12. Viguerie, A.; Lorenzo, G.; Auricchio, F.; Baroli, D.; Hughes, T.J.R.; Patton, A.; Reali, A.; Yankeelov, T.E.; Veneziani, A. Simulating the spread of COVID-19 via a spatially-resolved susceptible–exposed–infected–recovered–deceased (SEIRD) model with heterogeneous diffusion. *Appl. Math. Lett.* **2021**, *111*, 106617. [CrossRef] [PubMed]
13. Finney, R.L.; Weir, M.D.; Giordano, F.R. *Thomas' Calculus*, 10th ed.; Addisson Wesley Longman: Boston, MA, USA, 2001.
14. Sanders, R. On convergence of monotone finite difference schemes with variable spatial differencing. *Math. Comput.* **1983**, *40*, 91–106. [CrossRef]
15. de Boor, C. *A Practical Guide to Splines (Rev. Ed.)*; Springer: New York, NY, USA, 2001.
16. Simos, T.E.; Tsitouras, C. Efficiently inaccurate approximation of Hyperbolic Tangent used as transfer function in Artificial Neural Networks. *Neural Comput. Appl.* **2021**. [CrossRef]
17. Tsitoura, A.C.; Tsitouras, C. Cubic spline approximation of transfer functions for speeding neural networks performances. *AIP Conf. Proc.* **2020**, *2293*, 420015.
18. Tsitoura, A.C. Design and Implementation of a Database with Interactions of Hearing Loss and Diabetes. Master's Thesis, Department of Electical and Computer Engineering, National Technical University of Athens, Athens, Greece, May 2020. (In Greek) [CrossRef]
19. Fritsch, F.N.; Carlson, R.E. Monotone Piecewise Cubic Interpolation. *SIAM J. Numer. Anal.* **1980**, *17*, 238–246. [CrossRef]
20. *MATLAB Version R2019b*; The Mathworks, Inc.: Natick, MA, USA, 2019.
21. Medvedeva, M.A.; Simos, T.E.; Tsitouras, C.; Katsikis, V.N. Direct Estimation of SIR model Parameters through Second Order Finite Differences. *Math. Meths. Appl. Sci.* **2021**, *44*, 3819–3826. [CrossRef]
22. Available online: https://raw.githubusercontent.com/datasets/covid-19/main/data/countries-aggregated.csv (accessed on 10 March 2021).
23. Wikipedia. Available online: http://en.wikipedia.org/ (accessed on 10 March 2021).